Section 32

Aircraft Type Data

Processing

The FAA prints aircraft type data in Publication 7340 *Contractions*, Chapter 5, and on tape as part of the 56 day cycle (see *Geographical Data* Section). Note that the format and contents of the tape data was changed as of October 1990, whereby the **acft_category** field (see Table 32-1), was eliminated. As a result, the aircraft type tape data is read, and changes from the previous file are manually edited into the source file **candidates_ac.** The source file itself is used by the *FDB* during its initialization; it is processed into two forms for use by the run-time ETMS. The *match_planes* function, which is part of the *Build Aircraft Dynamics Database* process, produces three of the seven map files which form the **aircraft dynamics database**, as well as a file of aircraft types, **aircraft_categories.map**; its data structure is shown in Table 32-1. **Aircraft_categories.map** is used by the *Schedule Data Base* (SDB) and the *Parser* functions to look up aircraft types, and to assign to the flight record categories such as weight class and usage, which is then passed along to the *Flight Database Processor*.

The *match_planes* function is described in Section 32.1.1. The *map_profiles* function of the *Build Aircraft Dynamics Database* process uses the aircraft type data with aircraft dynamics data defined in the program declaration and assignment statements to complete the **aircraft dynamics database**, which is used by the *Parser* and *Flight Database Processor* functions to model flight altitude profiles and times. The *map_profiles* function is described in Section 32.1.2.

Table 32-1: acft_cat_type Data Structure

<table>
<tr><th colspan="5">acft_cat_type</th></tr>
<tr><td colspan="2"><b>Library Name:</b> ttm_</td><td colspan="3"><b>Purpose:</b> This structure holds data for every known aircraft type. The aircraft type is looked-up and the associated data for that type is passed back to the user.</td></tr>
<tr><td colspan="2"><b>Element Name</b>: acft_cat.ins.pas</td><td colspan="3"></td></tr>
<tr><th>Data Item</th><th>Definition</th><th>Unit/Format</th><th>Range</th><th>Var. Type/Bits</th></tr>
<tr><td>plane_type</td><td>Type of aircraft (e.g.,B727, A4)</td><td></td><td>-</td><td>string4</td></tr>
<tr><td>user_category</td><td>Use for aircraft (e.g., commercial, general aviation, military)</td><td>enumerated type</td><td>-</td><td>usercat_t</td></tr>
<tr><td>category_class</td><td>Flight category</td><td>enumerated type</td><td>-</td><td>flight_regs_t</td></tr>
<tr><td>acft_type</td><td>Area of usage (e.g., land, helicopter, seaplane)</td><td>enumerated type</td><td>-</td><td>actype_t</td></tr>
<tr><td>acft_category</td><td>Type category (e.g., civilian jet, single piston prop, fighter jet)</td><td>enumerated type</td><td>-</td><td>ac_cat_t</td></tr>
<tr><td>acft_weight</td><td>Aircraft weight class (e.g, small, large, heavy)</td><td>enumerated type</td><td>-</td><td>ac_weight_t</td></tr>
<tr><td>max_altitude</td><td>Aircraft types average altitude</td><td>Currently not in use.</td><td>-</td><td>short</td></tr>
<tr><td>avg_velocity</td><td>Aircraft types average velocity</td><td>Currently not in use.</td><td>-</td><td>short</td></tr>
</table>

32.1 The Build Aircraft Dynamics Database Process

The *Build Aircraft Dynamics Database* process creates (or recreates after alteration) the component files of the **aircraft dynamics database**. There are seven static mapped files which comprise the **aircraft dynamics database** and each is described in detail in Section 32.1.2.

There are two functions involved: *match_planes* and *map_profiles*. The first creates the **dsg_matchup_table** map file, which is used to determine the ascent and descent profiles for a particular aircraft. It also creates the **template_names** and the **lookup_profile** map files, which are used to address the ascent and descent profiles. *Match_planes* also creates the **aircraft_categories.map** file, which is used by the *SDB* and the *Parser.* The second function, *map_profiles,* creates the four profile map files which complete the **aircraft**

dynamics database, namely **ascent_alt_map**, **ascent_dist_map**, **descent_alt_map**, and **descent_dist_map**.

32.1.1 The Match_planes Function

Purpose

The purpose of *match_planes* is to generate map files which comprise part of the **aircraft dynamics database**. These mapfiles are used to assign a flight profile based on a given aircraft type. The data flows of the *match_planes* function of the *Build Aircraft Dynamics Database* process are diagrammed in Figure 32-1.

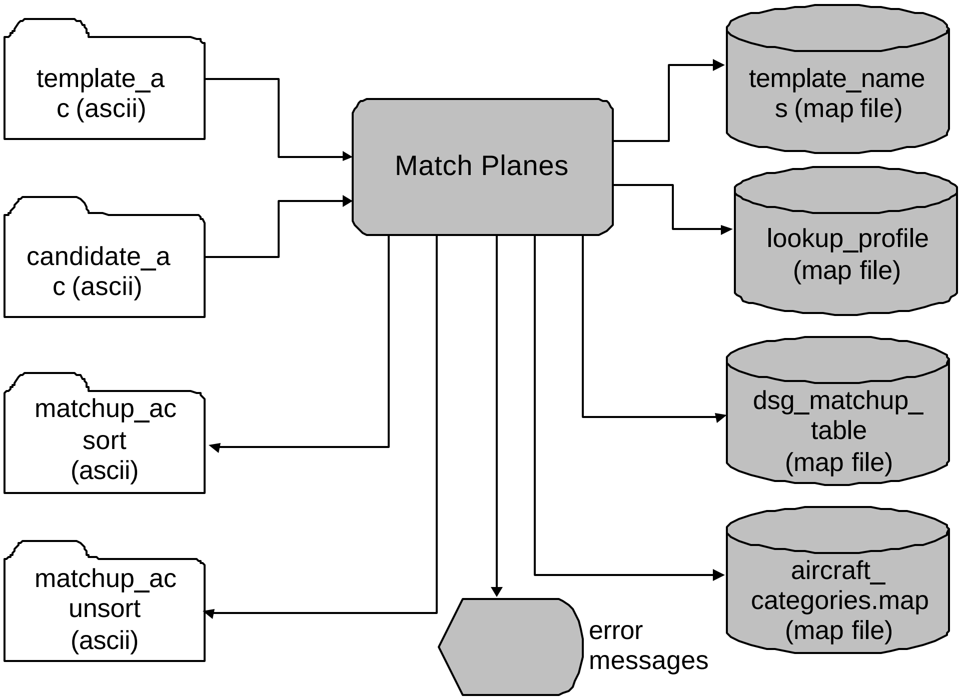
template_ac (ascii)
candidate_ac (ascii)
matchup_ac sort (ascii)
matchup_ac unsort (ascii)
Match Planes
error messages
template_names (map file)
lookup_profile (map file)
dsg_matchup_table (map file)
aircraft_categories.map (map file)

Figure 32-1. Data Flow of the Match_planes Function

Execution Control

The **aircraft dynamics database** map files are (re)created in a batch process by the ETMS operator, which requires executing both of the functions.

Input

Two ASCII files are needed to create the **dsg_matchup_table** map file, namely **template_ac** and **candidate_ac**. Each line of the **template_ac** file describes one of the 44 popular aircraft for which detailed profiles have been established. The **candidate_ac** file, sorted on the aircraft designator, contains a description of 704 aircraft types extracted from FAA Publication 7340 "Contractions", ATO-210. It also contains 11 aircraft types with Official Airline Guides (OAG) designators, because OAG does not seem to have a proper translation to FAA designators for these 11 aircraft types. At one time the structure of both files was similar; however, to consolidate functions and processes, the structure of **candidate_ac** was considerably modified. The structure of file **template_ac** is described in Table 32-2; the structure of file **candidate_ac** is described in Table 32-3. The nine aircraft category values in the **cat** field are described in Table 32-4; they are the same as the **acft_category** field of the **acft_cat_type** data structure, which is shown in Table 32-1. The first eight **used** to be defined by Publication 7340 (but are not anymore); the ninth, i.e., helicopters, was added to broaden the **aircraft dynamics database**.

The input data for both the **template_names** and the **lookup_profile** map files are defined in the program declaration and assignment statements.

Table 32-2: Aircraft Type Template File

Template_ac Fields				
Directory Name:		**Contents:** This file contains all the aircraft models. If an aircraft type is not an exact match, it matched to the “nearest” model.		
File Names: Template_ac datafile				
Data Item	**Column No.**	**Definition**	**Range**	**Type of Data**
Record #1-44				
cat	1-2	Category Number	1…9	integer16
wt_cls	6	Weight class	S or L or H	char
eng_num	9	Number of engines	1…8	unsigned short
dsg	13-16	Aircraft type designator		string4
	19-37	Aircraft model name Not read or used		string19
	39-58	Acft manufacturer – Not read or used		string20
clm	59-63	Max. sea level climb rate (feet/minute)		short
dive	65-69	Max. sea level dive rate (feet/minute)		short

	71	Civilian or Military –Not read or used	C or M	char

Table 32-3: Aircraft Type File

Candidate_ac Fields				
Directory Name:		**Contents:** This file contains all the known aircraft types (the information of a military designator may be the same as that of the civilian aircraft), and all their performance specifications.		
File Names: Candidate_ac				
Data Item	**Column No.**	**Definition**	**Range**	**Type of Data**
Record #1-n				
dsg	1-4	Aircraft type designator		string4
wt_cls	6	Weight class	S or L or H	char
mil_civ	8	Civilian or Military	C or M	char
lash	10	Land/Amphibian/ Seaplane/ Helicopter	L or A or S or H	char
cat	12	Category number	1…9	short
eng_num	14	Number of engines	1…8	short
power	16	Powerplant	P or T or J	char
clm	18-22	Max. sea level climb rate (feet/minute)		short
dive	24-28	Max. sea level dive rate (feet/minute)		short
	32-51	Aircraft model name Not read or used		string20
	53-72	Acft manufacturer -- Not read or used		string20

Table 32-4: Acft_category or cat Values

Value	Enumerated Name	Definition
0	UNK_AC_CAT	Unknown type.
1	SINGLE_PISTON_PROP	Single engine piston propeller
2	MULTI_PISTON_PROP	Multiple engine piston propeller

3	SINGLE_TURBO_PROP	Single engine turboprop
4	MULTI_TURBO_PROP	Multiple engine turboprop
5	CIVILIAN_JET	Large commercial turbojet
6	FIGHTER_JET	Attack/fighter turbojet
7	LARGE_MILITARY_JET	Large Military turbojet (cargo/tanker/bomber, etc.)
8	SPECIAL_TURBOJET	High performance turbojet
9	HELI_COPTER	Helicopter (of any type)

Output

The primary output from this function consists of three of the seven mapped files that comprise the **aircraft dynamics database**, and the auxiliary **aircraft_categories.map** map file. Secondarily, *match_planes* produces the two ASCII files **matchup_ac_unsort** and **matchup_ac_sort**, which mimic the contents of the **dsg_matchup_table** map file. (At one time, the difference between the two files was that the second file was sorted over aircraft designators; however, the input file **candidate_ac** is now sorted over designators, so the name difference is no longer appropriate. Rather the two files have different fields, and in a different order). The ETMS operator can use these output files to examine the results of the map files creation. *Match_planes* writes validation/error messages to the display screen for the operator as it progresses.

The first field of the **aircraft_categories.map** file contains the number of aircraft types that are in the file. The number is written to the map file so that any process which accesses the map file will know how many plane types are stored in the file. This ensures that no code will have to be re-compiled, if the number of aircraft type in the file is changed. A number of internal validation checks are performed, and warnings are generated if it appears that the input data (from **candidate_ac**) is not valid.

Processing

Match_planes first opens the input and output ASCII files. To create the **dsg_matchup_table map** file, the function reads each line of the **template_ac** file, sets an **ID** value (i.e., a template index) and computes the value of the **grp** variable using the values of **cat** and **wt_cls** listed in Table 32-5. This data is kept for each template in a temporary record, and all records are stored in arrays according to the **grp** values.

Table 32-5: Rules for Creating the grp Value

cat		wt_cls		grp
1 or 2				pistonprop
3 or 4				turboprop

5 or 8	and	(S or L)		large_com_jet
5 or 8	and	H		heavy_com_jet
6				fighter
7				big_mil_jet
9				helicoptr

Match_planes then reads each line in the file **candidate_ac**, translates the character input variable (i.e., **mil_civ**) into the Boolean output variable (i.e., **civ**), and then computes a **grp** value. The function uses the designator-template matching rule (described in Section 23.1.2.2.1) to assign an **ID** value to each candidate aircraft. The data for each candidate aircraft is stored in memory in the record structure. The data are then sorted using a shell sort algorithm and written out as a mapped file (i.e., the **dsg_matchup_table** map file). This same data is also written to the ASCII **matchup_ac_unsort** and **matchup_ac_sort** files for examination purposes.

The aircraft model names for the **template_names** map file and the profile indices for the **lookup_profile** map file are defined in the program declaration and assignment statements. The data structures are described in Sections 23.1.2.2.2 and 23.1.2.3.1 respectively. The data is written out into map files using the **ms_$crmapl** routine, which is supported by the operating system.

Error Handling

In general, messages describing any errors that occur during the batch processing are displayed on the ETMS operator's screen.

32.1.2 The Map_profiles Function

Purpose

The purpose of *map_profiles* is to generate the remaining map files that comprise the **aircraft dynamics database**. These map files are used to assign a flight profile based on a given aircraft type. The data flows of the *map_profiles* function of the *Build Aircraft Dynamics Database* process are diagrammed in Figure 32-2.

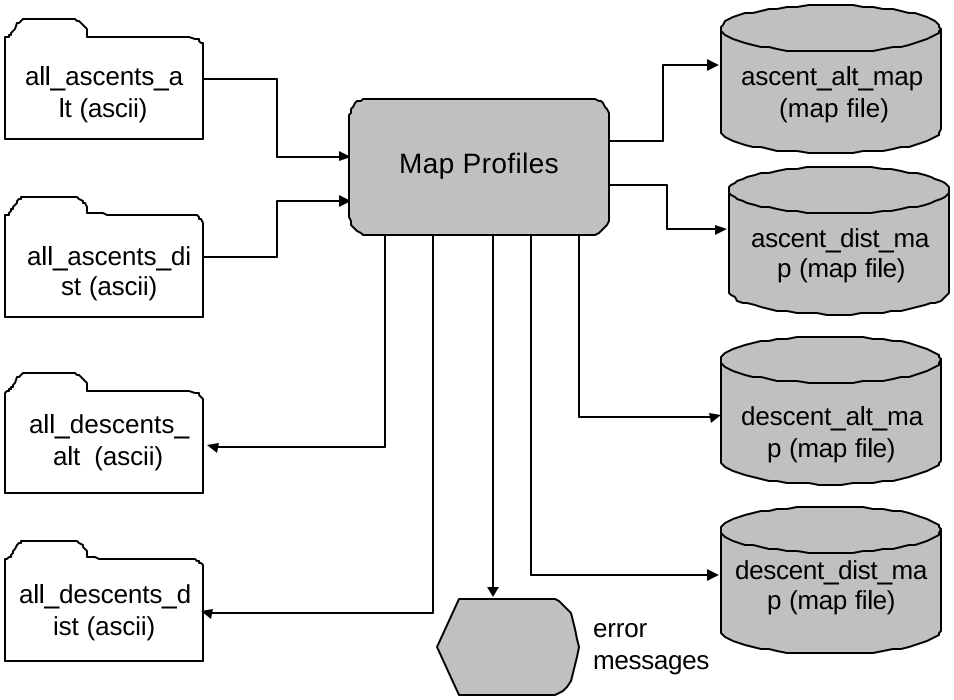

Figure 32-2: Data Flow of the Map_profiles Function

Execution Control

The **aircraft dynamics database** map files are (re)created in a batch process by the ETMS operator, which requires executing both of the functions.

Input

Four ASCII files are needed to create the four profile map files. Each input file is isomorphically mapped onto a profile map file as follows:

all_ascents_alt ⟶ **ascent_alt_map**,

all_ascents_dist ⟶ **ascent_dist_map**,

all_descents_alt ⟶ **descent_alt_map**, and

all_descents_dist ⟶ **descent_dist_map**

(1) The **all_ascents_alt** file has 7808 records. Each record contains one distance value (for that altitude level). All values in the file are formatted in **INT32**. The

file can be broken up into 61 record intervals, each of which is associated with one of 128 profiles.

(2) The **all_ascents_dist** file has 160128 records. Each record contains an altitude, a speed, and a time value (for that two-mile distance interval). All of these values are formatted in **INT32**. The file can be broken up into 126 record intervals; each interval is associated with one of 128 profiles.

NOTE: The speed is in units of 100*(nautical miles/minute). All the other elements are in their usual units.

(3) The **all_descents_alt** file has 61 records for that one descent profile. Each record contains one distance value (for that altitude level).

(4) The **all_descents_dist** file has 909 records. Each record contains an altitude, a speed (see note above), and a time value (for that two-mile distance interval). All of these values are formatted in **INT32**. The file can be broken up into 101 record intervals, each of which is associated with one of nine profiles.

Output

The primary output from this function consists of the remaining four map files that comprise the **aircraft dynamics database**. *Map_profiles* writes validation/error messages to the display screen for the operator as the function progresses.

Processing

In general, the **aircraft dynamics database** is kept in memory. If data in one or more map files are lost, or if it is known to be insufficient or incorrect, the operator must first examine the appropriate ASCII file and edit it, if required. When the ETMS is down, the operator executes *match_planes* and/or *map_profiles.*

The *map_profiles* function is straightforward. Each ASCII input file is read according to the Profile Maps data structure (described in Section 23.1.2.3, and written out using the **ms_$crmapl** routine, which is supported by the operating system.

All the input files are opened. A line of data is read in; the data is extracted from the appropriate fields and stored in memory in arrays that correspond to the map file structure. The process of reading a data line, extracting the data, and storing that data in memory is continued until all the data lines from the input file have been read. The map file is then written out, and a subset of the data from the map file is printed out. The function terminates when all four input files have been processed.

Error Handling

In general, messages describing any errors that occur during the batch processing are displayed on the ETMS operator's screen.